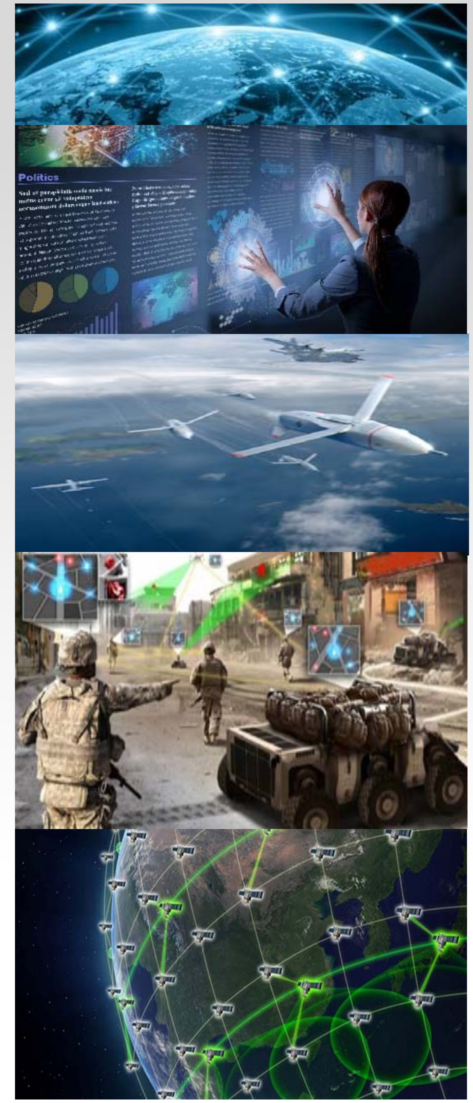
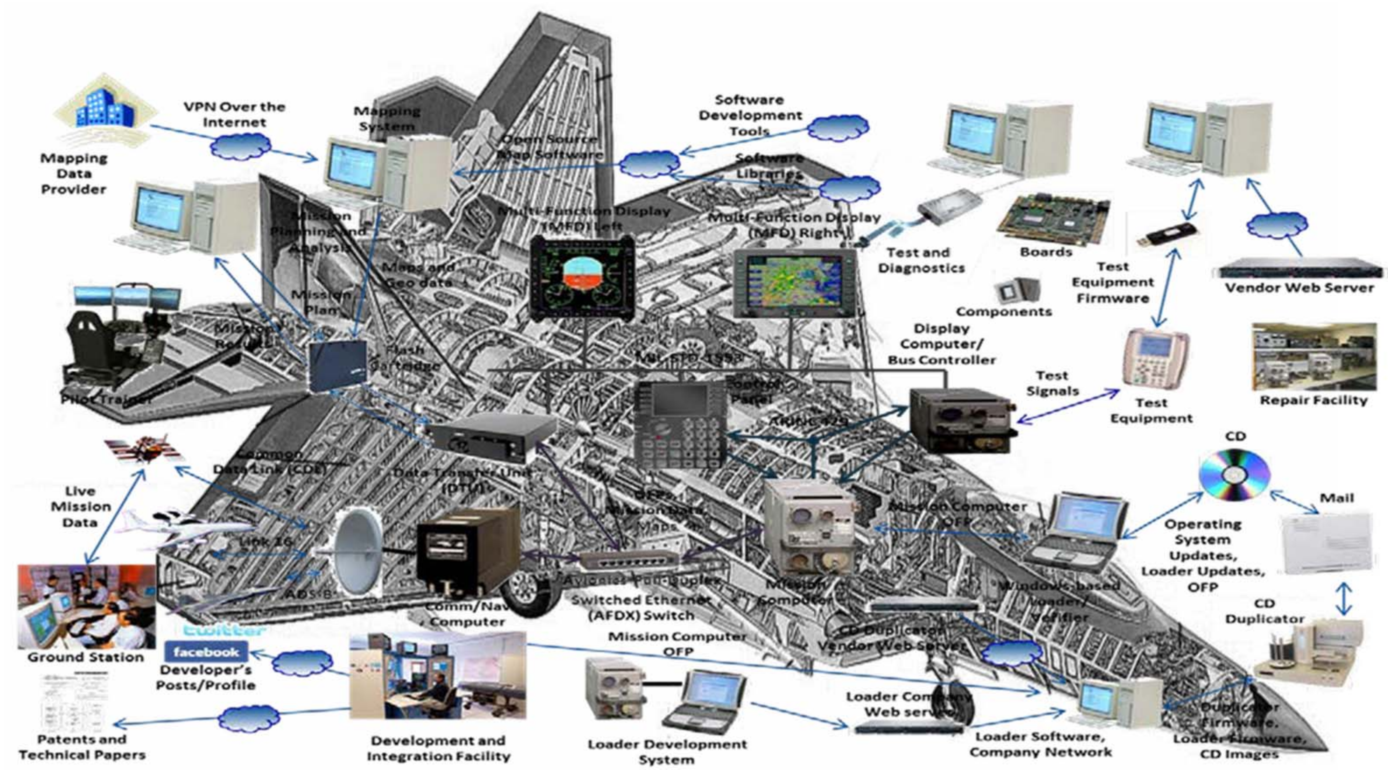
Software Assurance Throughout the System Life Cycle

Mr. Thomas Hurt
Strategic Technology Protection and Exploitation
Office of the Under Secretary of Defense for Research and Engineering

22nd Annual NDIA Systems and Mission Engineering Conference
Tampa, FL | October 23, 2019



Securing Complex Systems



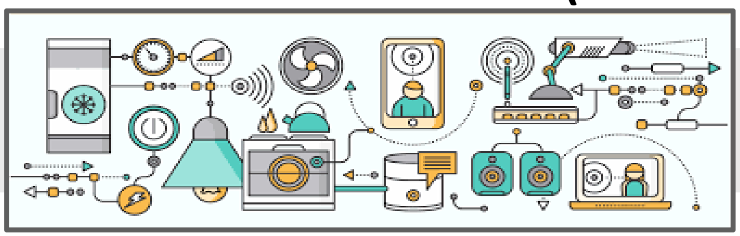
Modern military systems consist of hundreds of components with thousands of interconnections executing millions of lines of software.

Is the Future Sustainable?

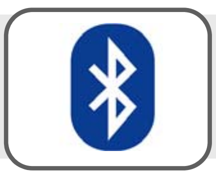
New Features/Components added continuously

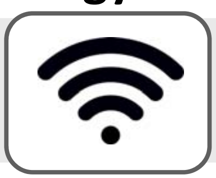
Components

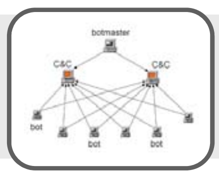
Everything is interconnected or networked (Internet of Things (IoT)



Technology continues to advance (methods of attack)





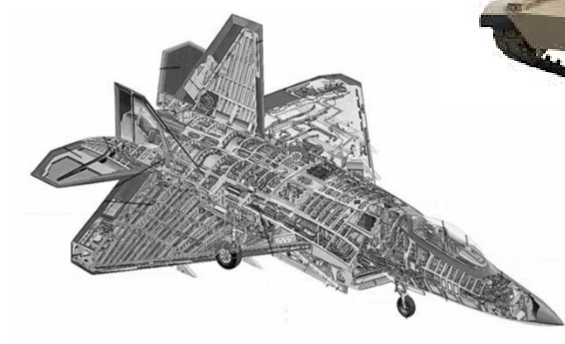


The addition of new components, changes to the network, and advancement of adversary technology creates an unsustainable continuous cycle of redesign and patching to protect against adversarial access.

Are We Really Protecting?

Scope:

Modern military systems consist of hundreds of components with thousands of interconnections executing millions of lines of software.



Cybersecurity:

The Risk Management Framework (RMF) is used to determine if a program receives an Authority to Operate (ATO) for a single version and configuration at the time of deployment.

Cybersecurity compliance is insufficient to address the scope of the threat.

Flaws in network architecture, mission software system design, and operations expose existing software weaknesses and known vulnerabilities to the adversary. Any of these weaknesses, when used by the attacker, can lead to degraded confidence, loss of lethality, or complete mission failure.

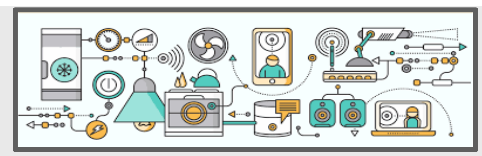
What If We Fixed The Vulnerabilities First?

New Features/Components added continuously

- Integrating new software, fixing known vulnerabilities first, is less challenging due to a reduction in exposure of vulnerabilities even when interoperating with vulnerable systems

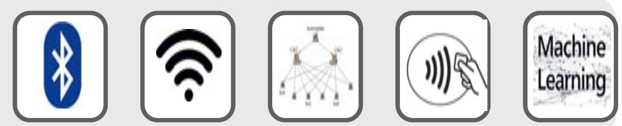
Everything is interconnected or networked (Internet of Things (IoT)

- In the event of unknown or unauthorized connection to secure systems, adversary remains unable to exploit vulnerabilities in connected devices.



Technology continues to advance (methods of attack)

- Removal of vulnerabilities prevents exploitation regardless of the adversary or attack method.



When DoD programs fix vulnerabilities, successful future cyberattacks are prevented. Instead of attempting to mitigate risk at the entry point, the vulnerabilities adversaries typically exploit have actually been removed.

Engineering Cyber Resilient Systems

System Concept Update

Updated System Tactical Use Threads

How will a component be used? Its tactical criticality?

Engine Control SW (ECS) provides needed metrics

Input: Engine performance data; Output: Needed alerts/response

Read/write capabilities to data bus do needed functions

Mission Threads

What will my system do and how will it interact?

Engine functionality will be controlled by ECS

Engine Monitoring System will monitor engine performance

Performance issues will be transmitted by data bus to control panel

System Requirements

What is required to get from concept to product?

ECS has no known vulnerabilities

Monitoring SW cannot be exploited to access ESC or data bus

Secure Design/Architecture considerations for Data bus communication

Countermeasure Selection

Tactical criticality focuses review of NVD and vendor SwA activities for COTS Binary Analysis

Static & Origin Analysis

Intel & Attack Modeling

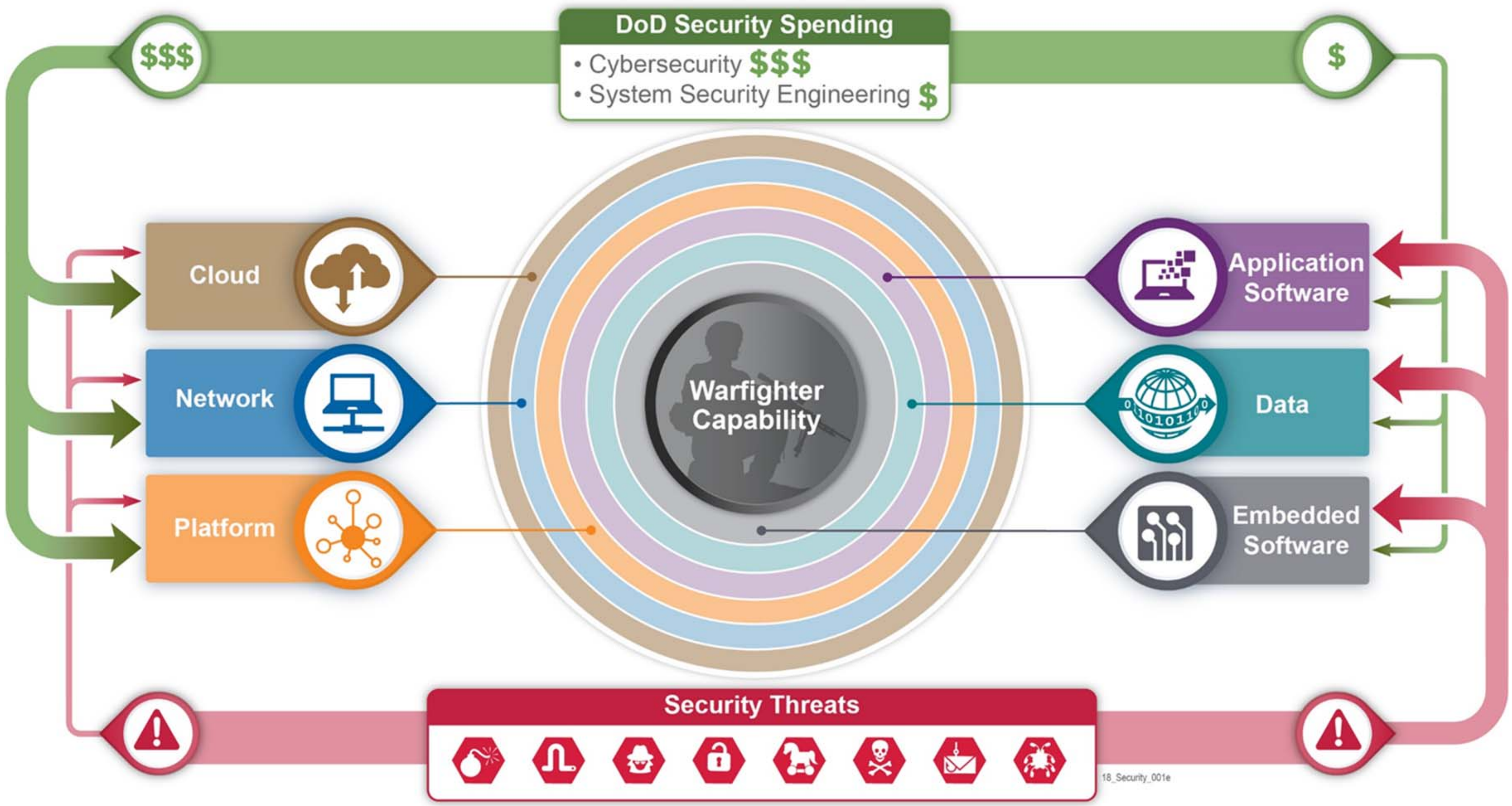
Coding Standards

Penetration Testing

Architecture/Design Inspections

Monitored Execution

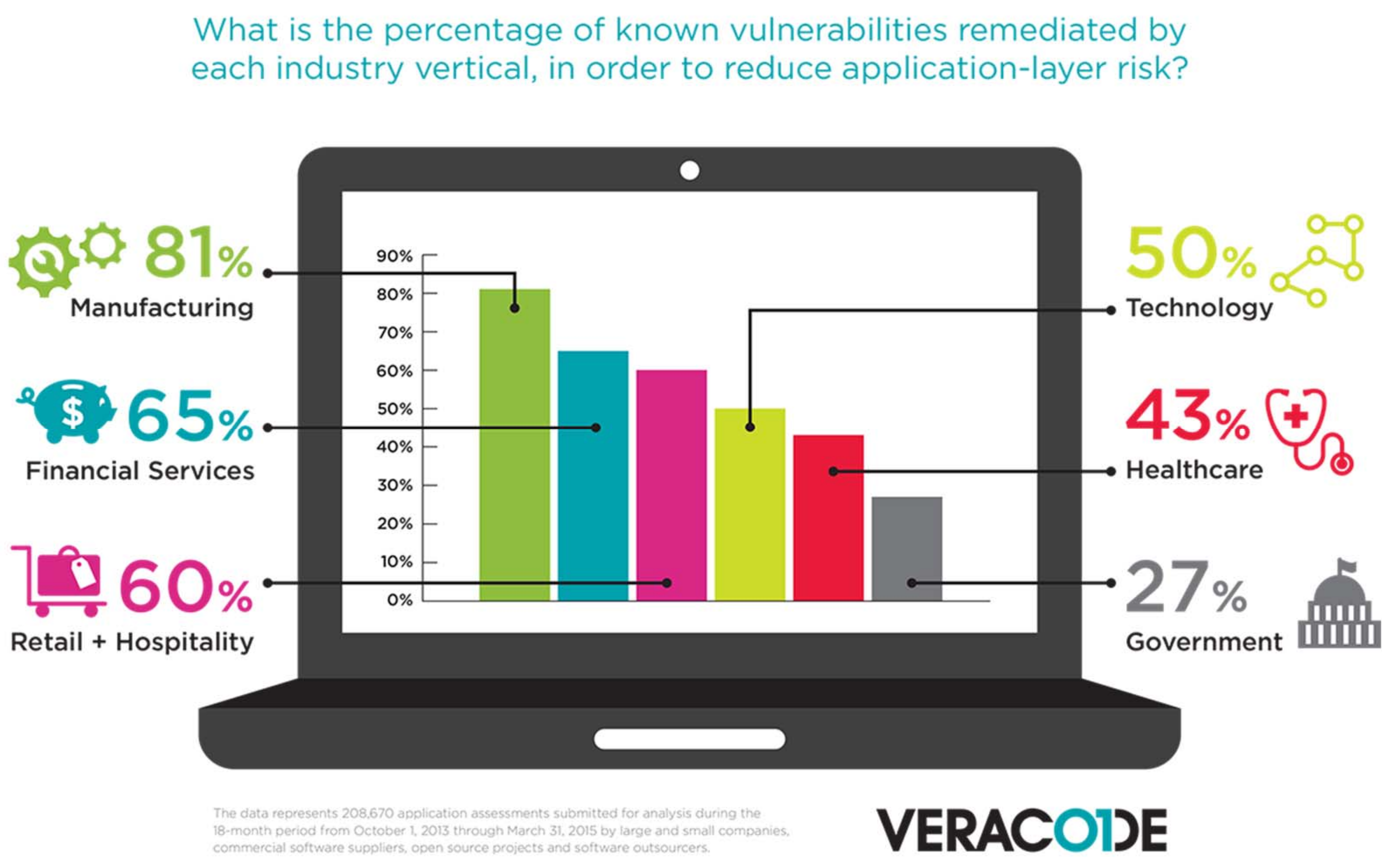
Department of Defense Security Spending



84% of breaches exploit the vulnerabilities in the application, yet funding for IT defense vs. software assurance is 23 to 1.

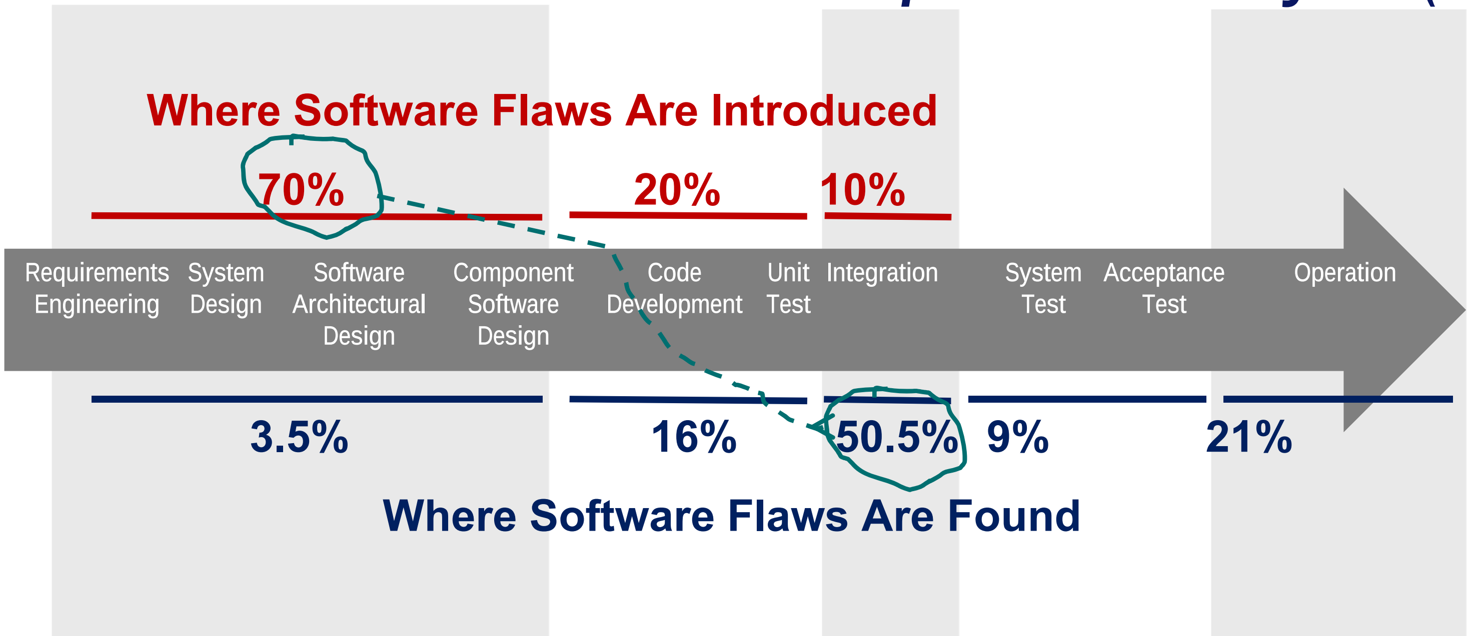
https://www.theguardian.com/technology/2016/oct/22/cyber-attack-hackers-weaponised-everyday-devices-with-malware-to-mount-assault "Baby Monitor Attack"

Who Fixes the Most Vulnerabilities?



Source: Veracode, used with permission: https://www.veracode.com/blog/2015/07/what-state-software-security-2015.

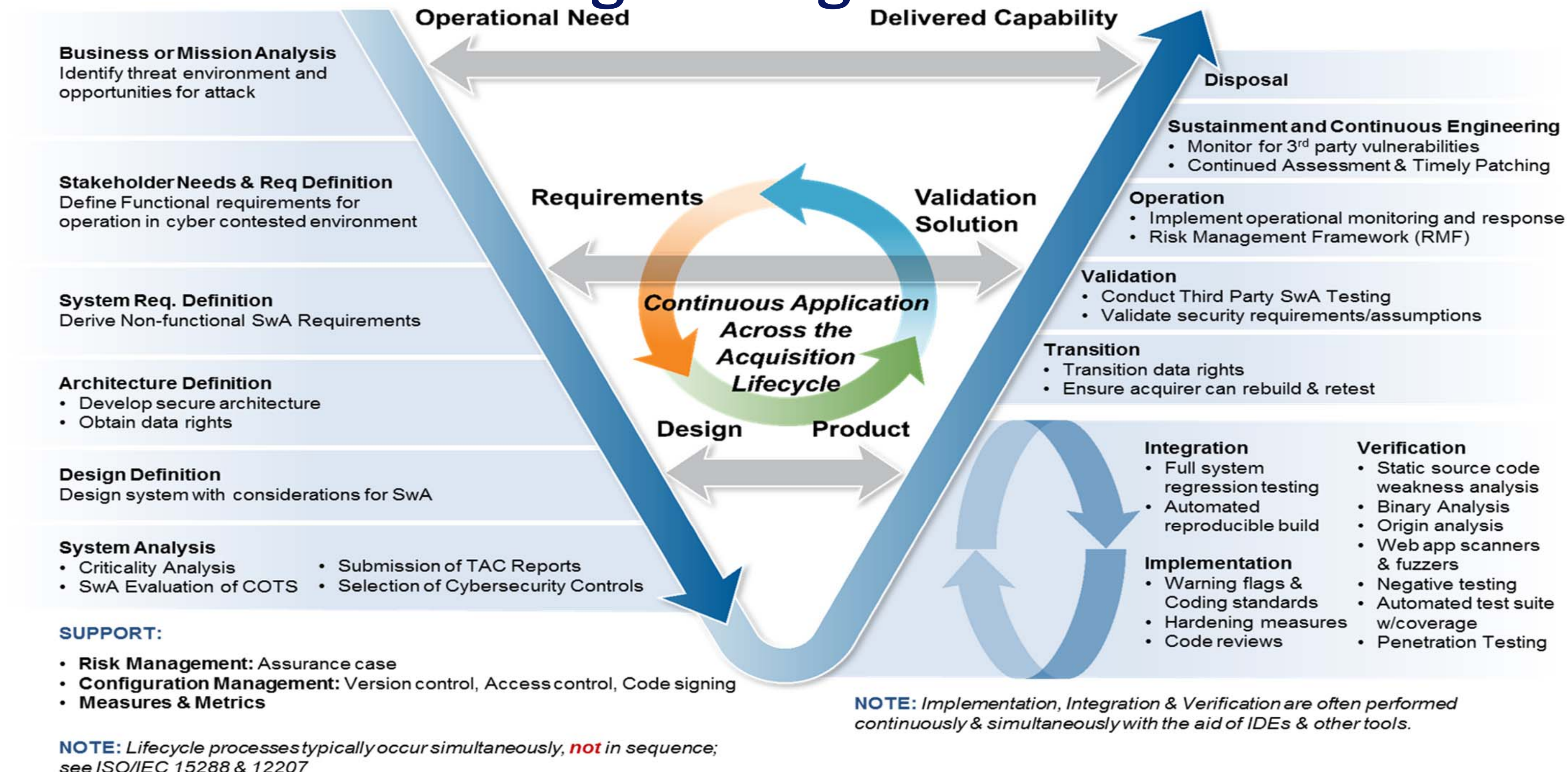
Contest: Need for Engineering-in Software Assurance Activities over the Software Development Life Cycle (SDLC)



Improved focus on engineering-in software assurance activities needed on the front end of the SDLC

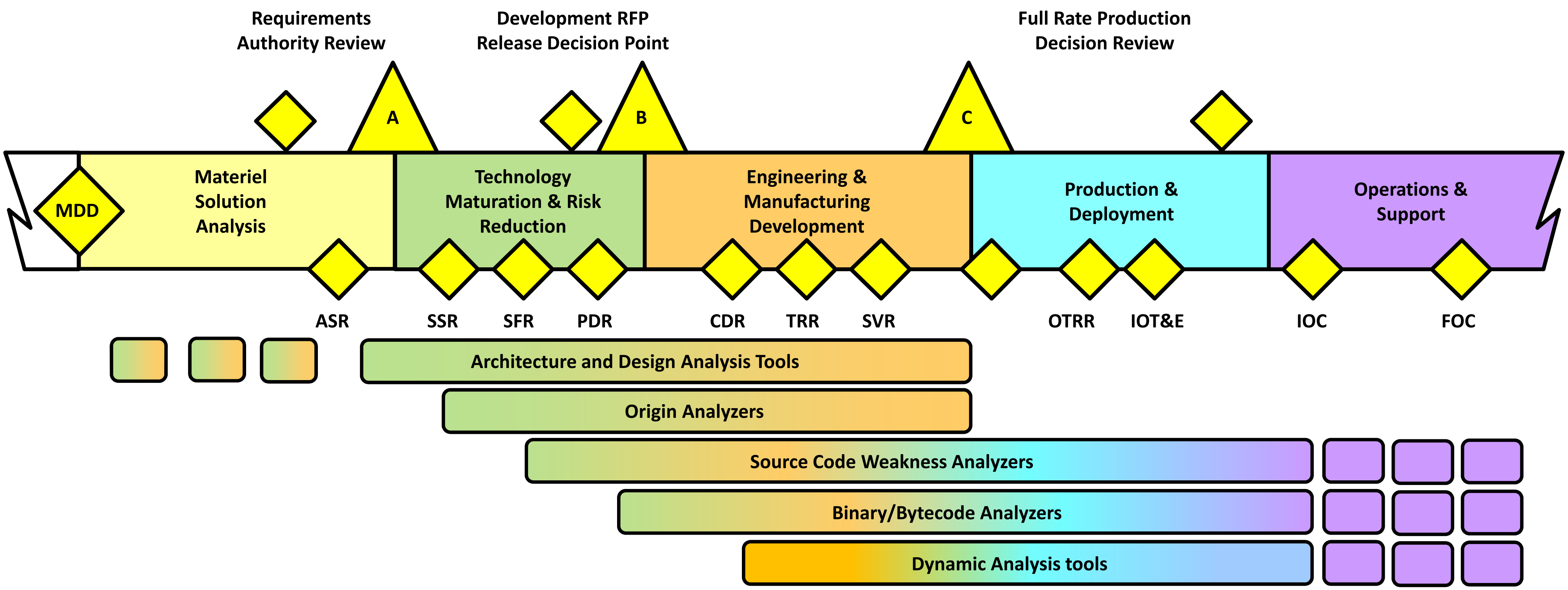
Source: Carnegie Mellon University, Software Engineering Institute *(Critical Code*; NIST, NASA, INCOSE, and Aircraft Industry Studies), used with permission.

Software Assurance Supplied to Traditional Systems Engineering Process



Distribution Statement A: Approved for public release; case # 20-S-0079 applies. Distribution is unlimited.

Use SwA Tools Throughout the System Life Cycle



With the integration and automation of software assurance tools throughout the system life cycle, programs can make informed decisions on the identification and mitigation of risk.

Distribution Statement A: Approved for public release; case # 20-S-0079 applies. Distribution is unlimited.

MDA Software Assurance Approach

Phase 1 SwA Pilot Program in partnership with OSD

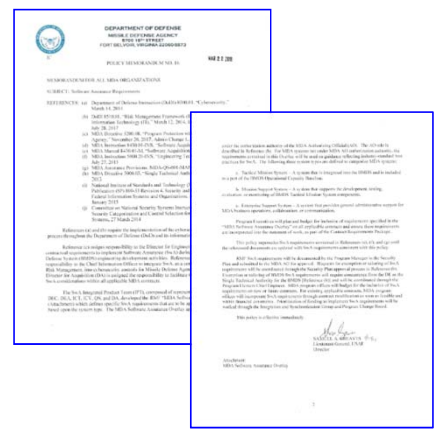
- Partnered with **DoD Joint Federated Assurance Center (JFAC)** Service Providers for SwA Risk Assessment Pilot Program
- Collaborated with **Carnegie Mellon University Software Engineering Institute** (CMU/SEI) to identify SwA Gaps in existing policy and guidelines

Phase 2

- Developed **SwA Contract Language, MDA Policies and Guidelines** based on identified gaps, **BMDS Software Threat Models** to support threat analysis and **SwA Training** for MDA personnel

Phase 3

- As of 22 March 2019, all MDA Programs under authority of the MDA Authorizing Official (AO) begin planning, budgeting, and incorporating SwA requirements into contracting efforts.



MDA Software Assurance Approach (cont.)

Ensure SwA requirements are on contract

- MDA Risk Management Framework enabled by SwA Overlay

Include SwA-specific CDRLs in ASSIST Database

- Software Assurance Evaluation Report
- Software Attack Surface Analysis Report
- Software Threat Analysis Report
- Vulnerability Assessment Report

Verify developer key SwA practices at milestone reviews

- SwA Entrance and Exit Criteria Added
  - Systems Requirements Review (SRR)
  - Preliminary Design Review (PDR)
  - Critical Design Review (CDR)
  - Test Readiness Review (TRR)

Control	Title	Justification for Selection	TMS[1]	MSS[2]	ESS[3]
SA-3	System Development Life Cycle	Secure Software/Firmware Development Life Cycle	+E	+E	+E
SA-4	Acquisition Process	Software/Firmware Security Requirements	+E	+E	+E
SA-4(2)	Acquisition Process: Design / Implementation Information for Security Controls	Government Purpose Rights	+E	+E	+E
SA-10	Developer Configuration Management	Software/Firmware Configuration Management	+E	+E	+E
SA-11	Developer Security Testing and Evaluation	Software/Firmware Security Testing and Evaluation	+E	+E	+E
SA-11(1)	Developer Security Testing and Evaluation: Static Code Analysis	Static Code Analysis	+E	+E	+E
SA-11(2)	Developer Security Testing and Evaluation: Threat and Vulnerability Analysis	Software/Firmware Threat Analysis	+E	+E	+E
SA-11(3)	Developer Security Testing and Evaluation: Independent Verification of Assessment Plans / Evidence	Independent Internal Verification of Assessment Plans/Evidence	+E	+E	+E
SA-11(4)	Developer Security Testing and Evaluation: Manual Code Reviews	Manual Code Reviews	+E	+E	+E
SA-11(5)	Developer Security Testing and Evaluation: Penetration Testing / Analysis	Penetration and Fuzz Testing	+E		
SA-11(6)	Developer Security Testing and Evaluation: Attack Surface Reviews	Software/Firmware Attack Surface Analysis and Reviews	+E		
SA-11(7)	Developer Security Testing and Evaluation: Verify Scope of Testing / Evaluation	Verify Scope of Software/Firmware Testing and Evaluation	+E	+E	+E
SA-11(8)	Developer Security Testing and Evaluation: Dynamic Code Analysis	Dynamic Code Analysis	+E	+E	+E
SA-12	Supply Chain Protection	Software/Firmware Supply Chain Protection	+E	+E	+E
SA-15	Development Process, Standards, and Tools	Software/Firmware Development Process, Standards, and Tools	+E	+E	+E
SA-17	Developer Security Architecture and Design	Software/Firmware Security Architecture and Design	+E	+E	+E
SI-2	Flaw Remediation	Software/Firmware Flaw Remediation	+E	+E	+E
SI-3	Malicious Code Protection	Malicious Code Protection	+E	+E	+E
SI-7	Software, Firmware, and Information Integrity	Software and Firmware Integrity	+E	+E	+E

[1] Tactical Mission System
[2] Mission Support System
[3] Enterprise Support System

Note: A plus sign ("+") indicates the baseline RMF control requirements specified in NIST SP 800-53 applies. The letter "E" indicates that there is a control extension for the applicable system type contained in the MDA SwA Overlay that applies. The blank cell indicates that the control is not required.

Last Thoughts

- DoD systems continue evolution toward extreme reliance on software for execution of critical and tactical functionality.
- The traditional application of "cyber protection" to DoD systems is sufficient to mitigate or prevent 10-to-16% of recorded cyberattacks.[1]
- Systems security engineering ensures implementation of SwA tools and practices from the start, transitions assurance requirements into sustainment, and is fundamental to ensure lethality of our weapons systems while under cyberattack.
- Program's, organization's, and industry's integration of more rigorous and robust software assurance engineering into their systems engineering process will reduce vulnerabilities and field more secure, reliable, and resilient weapons systems.

[1] Examples listed on following slide for reference.

Sample of Reporting on Cyber Attacks

- Is poor software development the biggest cyber threat?" CSO Online, September 2, 2015 https://www.csoonline.com/article/2978858/is-poor-software-development-the-biggest-cyber-threat.html
- "Most Cyber Attacks Occur From This Common Vulnerability," Forbes, March 10, 2015 https://www.forbes.com/sites/sap/2015/03/10/most-cyber-attacks-occur-from-this-common-vulnerability/#af90ba57454d
- "Engineering Software Assurance into Weapons Systems During the DoD Acquisition Life Cycle," Journal of Cyber Security and Information Systems, Volume: 5 Number: 3, Special SwA Issue, November 2, 2017 https://www.csiac.org/journal-article/engineering-software-assurance-into-weapons-systems-during-the-dod-acquisition-life-cycle/

DoD Research and Engineering Enterprise

Solving Problems Today – Designing Solutions for Tomorrow










DoD Research and Engineering Enterprise	Defense Innovation Marketplace	Twitter
https://www.CTO.mil	*https://defenseinnovationmarketplace.dtic.mil*	*@DoDCTO*

For Additional Information

Mr. Thomas Hurt

Director, JFAC / Deputy Director, Software Assurance
Strategic Technology Protection and Exploitation

Office of the Under Secretary of Defense
for Research and Engineering

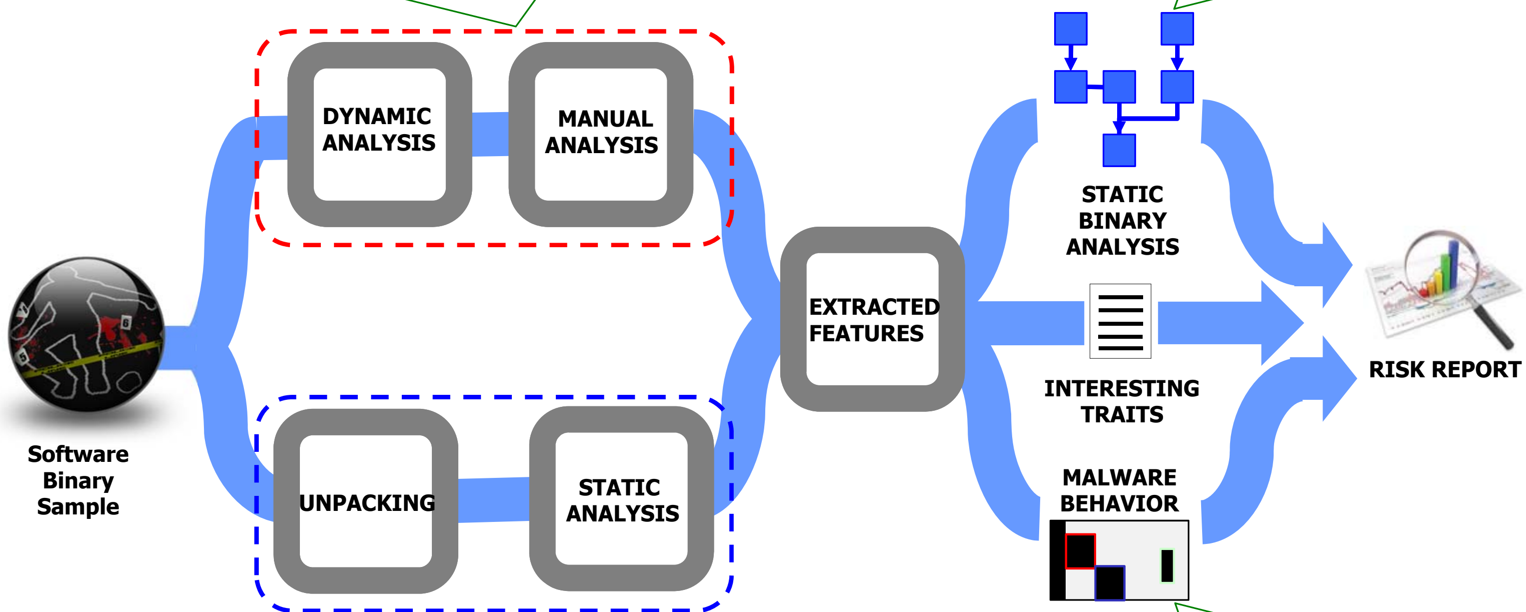
571-372-6129

thomas.d.hurt.civ@mail.mil

Technology: Binary Analysis Capabilities

Execution of binary samples in an instrumented virtual environment using a combination of automated and manual testing to identify potentially malicious code

Compute complete control flow and remove dead code that can be used for exploitation



Unpacking of software executable to run static test and examine structural features of the binary sample

Extract structuring and behavioral functions for analysis and include findings in risk report

*Source: AFLCMC TSN Lackland AFB

Program Protection Plan SwA Guidance

Report

Countermeasures	Design/Architecture Inspection	Coding Standards	Static Analysis	Origin Analysis	Independent SwA Testing	Version Control	Penetration Testing	Monitored Execution
CSCI 1	[illegible]	D	[illegible]		[illegible]		[illegible]	[illegible]
CSCI 2	A	D	D,I	I	V		V	O
CC/CF 1								O
CC/CF 2								O
COTS/GOTS 1						D,I,V		O
COTS/GOTS 2						D,I,V		O
FOSS 1		D	D,I			D,I,V		O
FOSS 2			D,I			D,I,V		

Add/remove columns to identify appropriate countermeasures based on the characteristics of the Program. Software assurance countermeasure activities should begin as early in the acquisition and system development life cycles as feasible and be implemented across all life cycle phases.

Per ISO/IEC/IEEE 15288, system development life cycle phases include:

Requirements Analysis (R) Architectural Design (A) Implementation/Development (D)
Integration (I) Verification & Validation (V) Operation (O) Maintenance (M)

Additional countermeasures for consideration:

Modeling and Simulation	Test Coverage Analyzer	Functional Architecture Assessment
Red Teaming	Attack Modeling	Formal Methods
Network Scanner	Network Sniffer	Fuzz Testing
Debugger	Intrusion Detection/Prevention Systems Problem Report Analysis Audit	
Processes	Logging Systems	
Tracked Data and Control Flow	ref. http://www.acq.osd.mil/se/docs/P-8005-SOAR-2016.pdf	

Report

Weaknesses									
	Total KSLOC[1]	KSLOC Assessed by Language[1]	Tools Used	Risk Level (Total/Mitigated) High	Risk Level (Total/Mitigated) Med	Risk Level (Total/Mitigated) Low	FRP Expectation (Total/Mitigated) High	FRP Expectation (Total/Mitigated) Med	FRP Expectation (Total/Mitigated) Low
CSCI 1									
CSCI 2									
COTS 1									
COTS 2									

Vulnerabilities									
	Risk Level			Risk Level (Mitigated/Patched)			Expectation at Fielding (Mitigated/Patched)		
	High	Med	Low	High	Med	Low	High	Med	Low
CSCI 1									
CSCI 2									
COTS 1									
COTS 2									

- What risk assessment methodology was used to determine risk level and how has the program prioritized the likelihood and consequence of vulnerabilities, identified in Table 4, for each critical software component? If risk method is the same as described in the Systems Engineering Plan, a reference is adequate.

Data Rights for SwA

Data Rights Requirements for Software Assurance

- Static analysis tools require source code
- Dynamic analysis tools require executable & environment
- Buildable source code & data rights required to have effective independent evaluation and competition of custom code.
- Need source code to be able to practically fix vulnerabilities (in most cases)

NDAA 2018 FY 2018 Sec. 871 Data Rights

As part of **any negotiation** for the **acquisition of noncommercial computer software**, the Secretary of Defense shall ensure that such negotiations **consider**, to the **maximum extent practicable, acquisition**, at the appropriate time in the life cycle of the noncommercial computer software, of **all software and related materials necessary**—

(1) to reproduce, build, or recompile the software from original source code and required libraries;

(2) to conduct required computer software testing; and

(3) to deploy working computer software system binary files on relevant system hardware.